Fault Modeling for Verilog Register Transfer Level

[1]Namita Palecha, [2]Poornima M and [3]Suma M S
[1, 3]ECE Department, RVCE, Bangalore, India
[2]ECE Department, MVJCE, Bangalore, India
namitapalecha@rvce.edu.in; poorni_m@rediffmail.com; sumams@rvce.edu.in

***Abstract* – As the complexity of Very Large Scale Integration (VLSI) increases, testing becomes tedious. Currently fault models are used to test digital circuits at gate level or at levels lower than gate. Modeling faults at these levels, leads to increase in the design cycle time period. Hence, there is a need to explore new approaches for modeling faults at higher levels. This paper proposes fault modeling at the Register Transfer Level (RTL) for digital circuits. Using this level of modeling, results are obtained for fault coverage, area and test patterns. A software prototype, FEVER, has been developed in C which reads a RTL description and generates two output files: one a modified RTL with test features and two a file consisting of set of test patterns. These modified RTL and test patterns are further used for fault simulation and fault coverage analysis. Comparison is performed between the RTL and Gate level modeling for ISCAS benchmarks and the results of the same are presented. Results are obtained using Synopsys, TetraMax and it is shown that it is possible to achieve 100% fault coverage with no area overhead at the RTL level.**

***Keywords* - fault coverage, fault models, fault simulation, stuck-at fault, RTL.**

I. Introduction

Rapid advances in technology are resulting in increasing the complexity of VLSI circuits. Performance, area, power and testing are the important attributes of any complex VLSI circuit. These attributes are directly affected with increase in the complexity of the circuits. This paper concentrates on only one of these attributes, i.e., testing.

Test generation is a process of finding input test patterns for detecting possible faults in the circuit. It is difficult to generate tests for real defects due to the diversity of VLSI defects. For generating and evaluating a set of test patterns, fault models are needed. Existing gate level fault modeling and fault simulation techniques exhibit poor performance standards when applied to such VLSI circuits and are unsuitable for early testability analysis or fault simulations [1]. These test generation and fault simulation efforts in the post synthesis phase do not contribute to the improvement in the design. Therefore, modeling faults and performing fault simulation at a higher level of abstraction is highly desirable.

Many high-level fault models and fault simulation techniques have been proposed [2]. However, none of the fault models are universally acceptable since no fault model has been developed so far that comprehensively covers all classes of faults in the circuits.

This paper explores fault modeling, fault simulation and test generation at the RTL level. The RTL description is at a higher level of abstraction and is described using a hardware description language such as Verilog. This paper also compares the RTL level and gate level circuits, in terms of fault coverage, test patterns and area.

Section 2 gives a brief background of testing at RTL. Section 3 describes the fault modeling. The proposed test methodology is given in Section 4. Section 5 provides the results. Finally Section 6 concludes the paper.

II. Background

Several researchers have been exploring fault modeling and testing at the RTL level [4, 5, 6, 7, 8, 9]. The fault model proposed by [4, 5] focused on Observability Enhanced Statement Coverage Metric to model faults. In addition to this metric [4] also uses single stuck at fault on all assignments targets of the executed statements. However, this fault model still does not cover all the faults associated with RTL description. In [5] the model requires that all statements in the RTL description are executed at least once and that their effects are propagated to at least one primary output. While this approach can be effectively exploited for the test pattern generation, more accurate results are needed for fault modeling.

In [6], the single stuck at faults are mapped to the behavioral domain. A functional analysis technique is used to evaluate the effects of the single stuck-line faults on gate-level implementations. The authors in [6] concluded that modeling all possible gate-level faults at the RTL is highly inefficient.

The RTL fault model and simulation approach proposed by [7] uses the single stuck-at fault for each bit of all variables in the RTL model. The model employs both the RTL description and functional verification patterns. The drawback of this approach is, it required one to run fault simulation twice; first in an optimistic mode and then in the pessimistic mode and to use the average of the results to reduce the difference between the RTL and the gate-level fault coverage. The experimental data shows that there is 10% error between the actual gate-level fault coverage and the RTL fault coverage.

In [8], probabilistic method for controllability evaluation based on a traitorously selection of registers to form groups is proposed. This work needs further optimization by computing the probabilistic impact of the simultaneous correction of different testability problems.

The fault model proposed by [2] focuses on fault modeling and simulation at RTL level and aims at exploring the capabilities of the stuck-at fault model in computing the fault coverage at RTL level. In this approach, for every conditional operation in the behavioral verilog, an error circuit is

DOI: 03.LSCS.2013.3.17

generated. For example if the verilog code has 3 conditional operators such as ==,! =, >=, then three error behavioral codes are generated and then each one is simulated. The results of this simulation are checked with the good simulation, to get the "detected" or "not detected" status of the fault. Though the approach provides 100% fault coverage, it is time consuming to generate error circuits for each condition and also for simulating. Furthermore, [2] does not consider stuck at faults in single bit values. The main focus of the current paper is to consider faults at all the ports. There is no need to generate error circuit for every condition hence reducing the fault simulation time.

III. Fault Modeling And Simulation At Rtl

Digital circuits are commonly designed at multiple levels of abstraction, including the layout, transistor, gate, register-transfer (RTL) and behavioral levels. Designers describe circuits in a hierarchical, top-down fashion, typically using computer-aided design (CAD) tools. To simplify the design process, designers try to model circuits at a fairly abstract level. Conventional gate-level implementation is hard to understand, i.e., poor readability. By modeling circuits at a higher level, the number of primitive elements in a circuit is reduced, thus making the problem size more tractable. This allows larger circuits to be handled in less time. The authors of [8] conclude that over 1000 times reduction in test-generation time is achievable by performing automatic test pattern generation (ATPG) at the RTL without any compromise in fault coverage.

Fault coverage is the total number of faults that are detected by using a set of test patterns. A variety of defects in a circuit; results in different types of faults. These types of faults are segregated to form various fault models. Fault models are defined at all the different levels of abstraction. Fault models are necessary for generating and evaluating a set of test patterns. A good fault model must provide the true behavior of the circuit in presence of all types of fault. Achieving such a model is difficult; hence a combination of different fault model is usually used in fault simulation and test pattern generation. This paper proposes to use single stuck at fault model at input and output ports of the RTL. Fault simulation and test pattern generation is done for all these faults at RTL. This RTL fault model is further mapped to the lower gate level.

In a single stuck at fault model in logic gate, either one of the inputs or outputs is stuck at logic0 (s-a-0) or logic1 (s-a-1). For a circuit with k input and output ports there can be 2k stuck-at faults and if n represents the number of inputs then the total test patterns required for detecting all the 2k faults is 2^n.

Lets consider an example of 2_to_1 multiplexer, the RTL description of which is shown in Figure1.

The fault model is considered at the RTL. Faults are considered at the input and the output ports. Buffers are inserted for each input and output port in the RTL code. The buffers inserted in the fault free circuit, should not disturb the functionality of the circuit. This insertion results in modified RTL, which can be treated as a faulty code for fault simulation.

```
module mux2_to_1 (s,a,b,y);
        input s;
        input a,b;
        output y;
        reg y;
always @(*)
        if(s) y=a;
        else y=b;
endmodule
```

Figure1. RTL description of 2_to_1 Multiplexer

Test patterns are generated to perform fault simulation for both the fault free and faulty module. Then the final fault coverage results are obtained. Fault coverage is the ratio of number of faults detected to the total number of faults in the module.

IV. Methodology

The proposed fault simulation and test patterns generation flow at RTL is compared with the standard approach at gate level. This is depicted in Figure 2.The path on the left is the proposed RTL flow and that on the right is the standard gate level flow. In order to access the inputs and outputs in the RTL, buffers are inserted at each port. These buffers do not affect the functionality of the circuit. The modified (faulty) verilog code is further used for analysis of fault coverage and test pattern generation. In the MUX example, let's assume input 'a' is stuck at 0. In order to detect this fault test pattern '110" is applied at the inputs. The expected output at 'y' is '1'. However in presence of stuck at 0 fault output 'y' will be at logic 0.

Methodology of inserting buffers in the proposed approach has been automated and a prototype has been developed in C, FEVER (Fault modEling for VErilog RTL). FEVER reads an RTL verilog description, parses it and generates the faulty RTL verilog code. The faulty code is then used by the prototype FEVER to generate the test patterns. An algorithm has been developed to reduce the number of test pattern, which in turn will aid in decreasing the test time. These reduced pattern set is then used to simulate and obtain the final fault coverage at RTL. The fault free RTL is used to obtain the final gate level netlist. In order to obtain the fault coverage for this netlist standard scan insertion is used. Fault simulation is performed on fault free and faulty circuit at both RTL and gate level. The output of the bad module is compared to that of good module to find the fault coverage.

DOI: 03.LSCS.2013.3.17

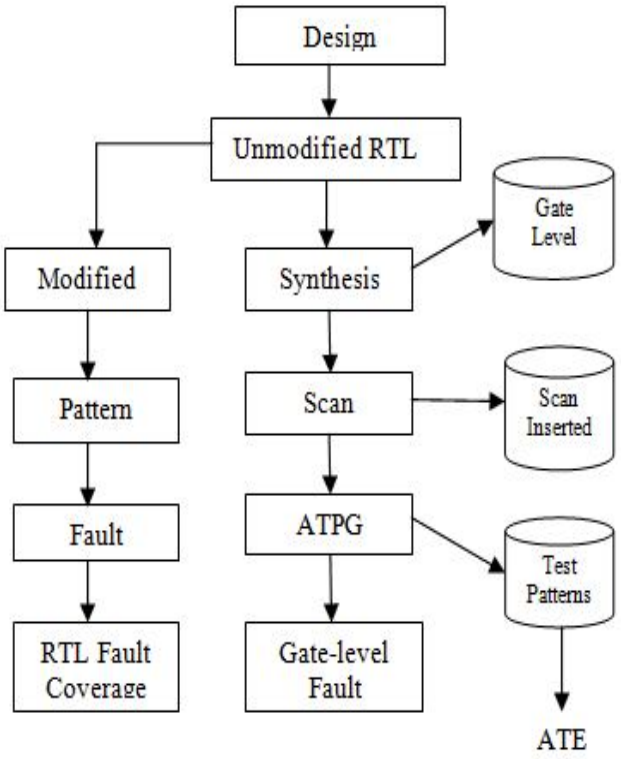

Figure.2. Design flow with the proposed method

V. Results

The proposed approach has been verified on the standard ISCAS benchmarks. FEVER was used to generate the faulty RTL and the corresponding test patterns. These patterns were further reduced using an algorithm and a test module written in verilog. This test module is used to obtain the fault coverage at RTL. The RTL code is synthesized to obtain final netlist using Synopsys Design Compiler. The netlist is mapped to the TSMC-65nm technology library. ModelSim and TetraMax are used to simulate and obtain fault coverage for both RTL and Gate level. The results of RTL and gate level are tabulated in Table1. Column1 are the ISCAS benchmarks. Column 2 and 3 give the total number of faults and reduce test patterns at RTL. The same figures for corresponding gate level are given in column 4 and 5 respectively. The last two columns depict the fault coverage at RTL and gate level. Figure 3 shows that number of faults and test patterns at RTL is significantly less compared to faults at gate level. Faults at RTL can be 100% mapped to gate level but the reverse is not possible [1]. Introducing DFT usually results in increase in area overhead. The results for this at RTL for ISCAS are given in Table 2. Inserting DFT at RTL does not impact the area as shown in columns 2 and 3 of Table 2. These DFT features after synthesis are mapped to zero delay buffers, hence not affecting the area. When synthesized using different technology libraries (TSMC-65nm, TSMC-180nm) these zero delay buffers are mapped to wires.

VI. Conclusion

With the progress of semiconductor technology testing of VLSI circuits is not easy. Fault simulation at gate or lower level, done after synthesis does not help in improving the circuit design. This would result in increasing the time and cost for redesigning the circuit. This paper introduces hence testing circuits at higher level. With the proposed approach RTL designers can have an estimation of the fault coverage before doing synthesis and also it is possible for the designer to locate faults at a higher level of abstraction. A prototype in C, FEVER has been developed to insert the DFT and also generate the test patterns. This prototype has been verified on ISCAS benchmark circuits. Future exploration involves detecting fault coverage at the gate level using the test patterns generated at RTL.

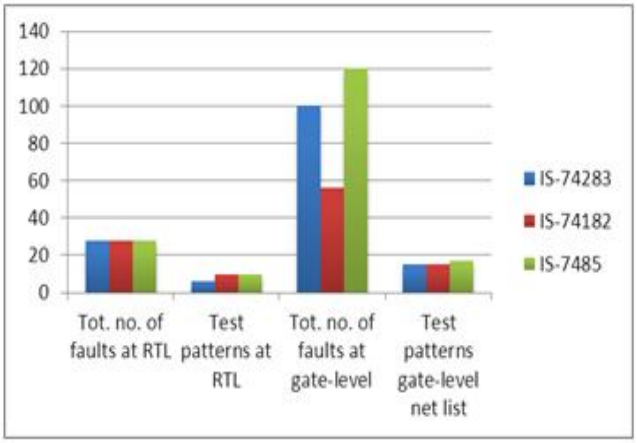

Figure.3 RTL and gate level Faults and test patterns comparison

Table II. Resultant Area

Benchmarks	RTL Area (nm)	
	Without DFT	With DFT
74283	50.4	50.4
74182	56.16	56.16
7485	56.2	56.2

References

[1] Suma M S, K.S. Gurumurthy " Fault Modeling of Digital Circuits at RTL" at International Conference on Circuits, System and Simulation (ICCSS), 2011.

[2] Karunaratne, Sagahyroon, Prodhuturi,"RTL Fault Modeling"IEEE Circuits and Systems August, 2005.

[3] J.Bhaskar, "Verilog HDL Synthesis, A Practical Primer", BSPublications, 2004.

Table I. RTL Versus Gate-Level Fault Coverage

Benchmarks	Total number of faults at RTL	Test patterns at RTL	Total numbers of faults at the gate-level	Test patterns at the scan inserted gate-level net list	RTL fault coverage	Gate-level fault coverage
74283	28	6	100	15	100%	100%
74182	28	10	56	15	100%	100%
7485	28	10	120	17	100%	100%

DOI: 03.LSCS.2013.3.17

[4] F.Corno, G.Cumani,M.Souxa Reorda,G.Squillero,"An RT-level Fault Model with High Gate Level Correlation "Proceedings of the IEEE International High_Level Design Validation Test Workshop.

[5] Ronald J Hayne and Barry W.Johnson,"Behavioral Fault Modeling in a VHDL Synthesis Environment", IEEE VLSI Test Symposium, 1999.

[6] Weiwei Mao, Ravi K Gulati,"Improving Gate Level Fault Coverage by RTL Fault Grading", IEEE Proceedings of the International Test Conference, 1996.

[7] Devadas, A.Ghosh, K.Keuter,"An Observability-Based Code Coverage Metric for Functional Simulation"Proceedings of IEEE/ACM International Conference on Computer Aided Design, 1996.

[8] Jose M.Fernandes,Marcelino B.Santos,Arlindo L.Oliveira,Joao C.Teixeira,"IEEE International High Level Design and Test Workshop,2006.

[9] L.T.Wang, C.W.Wu, X.Wen,"VLSI Test Principles and Architectures, "Morgan Kaufmann Publishers, 2006.

[10] Chen C.H,Noh T.H,"VHDL behavioral ATPG and fault simulation of digital systems,"IEEE transactions Aerospace and Electronic Systems,April 1998,pp 430-447.

[11] P.Goel, "Test Generation Cost Analysis and Projections," Design Automation Conference, 1980.

[12] Himanshu Bhatnagar," Advanced ASIC Chip Synthesis "Kluwer Academic Publishers, 2000.

[13] P.K.Lala,"Digital Circuit Testing and Testability, "Academic Press, 1997.

DOI: 03.LSCS.2013.3.17